Amendments to the Specification

Please rewrite the paragraph at page 2, lines 26 – 28, as follows:

Accordingly, it is desirable to provide techniques for improving the flexibility of network interfaces. Further, it is desirable to improve flexibility, while efficiently managing the resources in [[an]] a network interface.

Please rewrite the paragraph at page 4, lines 1 – 7, as follows:

The logic which dynamically allocates the memory maintains a list of buffer descriptors for corresponding buffers in said memory. The buffer descriptors include a parameter, which is programmable in one embodiment, that specifies the size of the corresponding buffer. Also, the buffer descriptors include a parameter that specifies a location of the corresponding buffer in the memory array. The list of buffer descriptors according to one embodiment comprises a free buffer list and a used buffer list for each of the virtual paths served by the system.

Please rewrite the paragraph at page 6, lines 5 – 10, as follows:

A detailed description of embodiments of the present invention is presented with reference to Figs. 1 through 15, in which Figs. 2-8 illustrate [[a]] an operation of the virtual paths with reference to an architecture suited to statically allocated memory. The structures for managing the plurality of queues according to priorities based on quality of service levels described in

Figs. 2-8 are used also in embodiments of the present invention more readily suited to use of dynamically allocated memory of the virtual paths.

Please rewrite the paragraph at page 8, line 26 – page 9, line 7, as follows:

The network interface chip 100 includes virtual path arbitration and control logic 110, and a timeout counter/register 111 supporting the second FIFO queue, and a timeout counter/register 112 supporting the third FIFO queue. The virtual path arbitration and control logic supplies signals muxSel1, muxSel2, muxSel3 on the 113 to the data multiplexer 106, in order to select data from one of the first, second and third storage arrays 121, 122, 123. Also the virtual path arbitration and control logic 110 supplies read pointers, vplRdPtr[9:0], vp2RdPtr[9:0], vp3RdPtr[9:0], on line 118 to the first, second and third storage arrays 121, 122, 123. The virtual path arbitration and control logic 110 is responsive to valid bits vp1Valid, vp2Valid, vp3Valid supplied on line 114 from the respective FIFO queues in the transmit packet buffer 120 which indicate that a valid packet is present in the respective queues. Also, the virtual path arbitration and control logic 110 is responsive to the outputs vp2Timeout, vp3Timeout on lines 115 and 116, respectively, of the timers 111 and 112. The virtual path arbitration and control logic 110 also controls [[of]] the reset and loading of the timeout timers 111, 112 via control signals vp3RdDone, vp2RdDone, vp2LdCnt, vp3LdCnt on line 117.

Please rewrite the paragraph at page 9, lines 8 – 15 page 9, as follows:

Fig. 3 shows the data structure for each virtual path in the transmit packet buffer 120. Thus, a FIFO queue includes contiguous storage space for a plurality of packets in this example. A first packet in the FIFO queue has a frame start leader 150 at a location pointed to by a read pointer 151 (RdPtr). The first packet has packet data 152, typically adjacent the frame start header 150. An end of packet pointer 153 (EopPtr) points to the address of the frame start header 154 of the next packet in the FIFO queue. The next frame includes packet data 156. A write pointer 155 (WrPtr) points to the next free address in the FIFO queue to which data is written as it is downloaded into the FIFO queue from the host.

Please rewrite the paragraph at page 10, lines 1 – 13, as follows:

The download DMA engine 108 is used for packet download from host memory. The packet goes through the PCI Bus 101 to the Transmit Packet Buffer (TPB) 120. The TPB 120 consists of three virtual paths which can be used to handle three quality of service levels, for example, real time traffic, normal traffic and IPSec traffic. Based on the quality of service level set in Frame Start Header (FSH), the packet is downloaded into the corresponding virtual path. The code wrData[28] is used to indicate real time traffic. When this bit it set, vp1We will be asserted to push the packet into virtual path 1 TPB. The code wrData[29] is used as an indication of the normal traffic. If this bit it set, vp2We will be asserted to load the packet into virtual path 2 TPB.

Finally, wrData[30] set will be used to handle IPSec traffic. The assertion of vp3We will be used to download an IPSec packet into virtual path 3 TPB. Packet download ~~started~~ starts with a FSH with an invalid packet indication followed by packet data write. At last, a final FSH will mark the downloaded packet to be valid with vpEopPtr stored to indicate where the packet ends.

Please rewrite the paragraph at page 14, lines 18 – 30, as follows:

In order to prevent starvation and guarantee forward progress, a preemption scheme is used to preempt higher priority virtual path traffic flow if lower priority virtual paths traffic are not serviced in time. This scheme will ensure that the higher priority traffic will be handled with minimum latency without stalling the lower priority traffic. Since virtual path 1 has the highest priority, no timeout counter for this virtual path will be needed. For handling virtual path 2 or virtual path 3 traffic flow, the corresponding virtual path timeout counter will be loaded and start counting down if there is any packet in its virtual path buffer (indicated by vp2Valid or vp3Valid). If the packet is transmitted out from the associated virtual path buffer before its time counts down to zero, the counter will be reloaded to its initial value and the above sequence will repeat. Otherwise, it indicates that the packet in the virtual path 2 buffer or the virtual path 3 buffer cannot be transmitted out in time. The vp2Timeout or vp3Timeout signals will be asserted to notify the virtual path arbitration and control logic 110 that preemption of higher priority traffic is needed to service lower priority traffic.

Please rewrite the paragraph at page 16, lines 15 – 21, as follows:

Fig. 9 provides a conceptual diagram of a system implementing the invention, including a host CPU 530 coupled to a PCI bus 501 and an integrated circuit 500 (such as ASIC 14 in Fig. 1) including the logical circuitry for transferring data packets into and out from a TPB 520 according to the present invention. The system includes a medium access control and physical layer circuit MAC/PHY block 502, [[,]] which is coupled to the network medium 503. For simplicity, the only parts of the circuit that relate to the host-to-LAN transfer (data transmit) direction are shown.

Please rewrite the paragraph at page 19, lines 5 – 24, as follows:

Fig. 12 shows the format of a used buffer descriptor 570. Each buffer descriptor 570 consists of buffer length in bits [6:0], buffer address in bits [22:7] and lastBuf indication in bit [31]. In this example, with a seven bit buffer length field, each buffer can be initialized up to 128 bytes in size. The buffer size can be selected to meet the needs of a particular implementation. The greater the buffer length is, the less buffer descriptors will be needed to support all the buffers within a virtual path. The potential drawback for larger buffer size is more space could be wasted within a buffer. This is true especially for smaller packets since each one may only occupy one buffer, without being aligned at a buffer boundary. If a packet consists of multiple buffers, the buffer length field for each buffer will be the maximum value

except the last one, which could be less if the packet is not ended at a buffer boundary. The buffer length field for the last buffer will be rewritten upon completion of the packet download operation to indicate where the packet ends. Buffer address is used to indicate the memory address for this buffer in the storage array used for the transmit packet buffer 520. This address can be used either to store the downloaded packet, or to read out the stored packet to be transmitted. Only the buffer address is needed for locating each buffer since the low-order bits can all be set to zeros pointing to the beginning of each buffer. A lastBuf bit is used to indicate the last buffer within a packet. This bit will be set in the last free buffer used to store the downloaded packet. While reading a packet out for transmission, a used buffer descriptor will be fetched and this bit will be set on the last used buffer, and is used by packet transmit engine 543 to determine where the packet ends.